International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-2 Number-4 Issue-7 December-2012

A review of Support Vector Clustering with different Kernel function for Reduction of noise and outlier for Large Database

Deepak Kumar Vishwakarma[1], Anurag Jain[2]
Department of Computer Science & Engineering, RITS, Bhopal, India [1, 2]

Abstract

For a long decade clustering faced a problem of noise and outliers. Support Vector Clustering is one of the techniques in pattern recognition. Support Vector Clustering is Kernel-Based Clustering. Division of patterns, data items, and feature vectors into groups (clusters) is a complicated task since clustering does not assume any prior knowledge, which are the clusters to be searched for. Noise and outlier reduces the mapping probability of sphere in support vector clustering. Support vector clustering is inspired clustering technique form the support vector Machine. The prediction and accuracy of support vector clustering depends upon kernel function of hyper plane. Kernel function is a heart of classifier. In this paper we present review of support vector clustering technique for pattern detection and reorganisation for very large databases. The variation of performance of support vector clustering depends upon kernel of classifier. Here we discuss different method of kernel used in support vector clustering.

Keywords

SVC, Kernel function, Outlier, SVM

I. Introduction

Clustering has always been a tricky task in pattern classification. Many clustering algorithms have been proposed in the past years. Division of patterns, data items, and feature vectors into clusters with different shapes, they still cannot produce arbitrary cluster boundaries to adequately capture or represent the characteristics of clusters in the dataset [1, 9]. A group (clusters) is a complicated task since clustering does not presume any prior knowledge, which are the clusters to be searched for. There exist no class label attributes that would tell which classes exist. Some of the traditional clustering techniques are Hierarchical clustering algorithms, Partitional clustering algorithms, nearest neighbor clustering, and Fuzzy clustering [5]. Clustering algorithms are capable of finding clusters with different shapes, sizes, densities, and even in the presence of noise and outliers in datasets. Although these algorithms can handle. Support Vector Clustering (SVC), which is inspired by the support vector machines, can overcome the limitation of these clustering algorithms. SVC algorithm has two main steps a) SVM Training and b) Cluster Labeling [39]. SVM training step involves construction of cluster boundaries and cluster labeling step involves assigning the cluster labels to each data point. Solving the optimization problem and cluster labeling is time consuming in the SVC training procedure [4]. Many of the research efforts have been taken to improve the efficiency of cluster labeling step. Only little work is done to improve the accuracy and efficiency of SVC training procedure. In recent time, specialists have made use of different cluster labeling techniques and different preprocessing procedures for improving the efficiency of SVC procedure. Preprocessing procedures used for SVC to reduce SVC training set are Heuristics for Redundant-point Elimination (HRE) and Shared Nearest Neighbor (SNN) technique result in loss of data. My main objective of research is to reduce the execution time of SVC procedure as well as to improve the ability of proposed SVC scheme in dealing with classification problems. SVC algorithm is to look for the smallest sphere that encloses the images of data points in the feature space. This sphere is then mapped back to the data space, where a number of contours which enclose the data points are formed. These contours are interpreted as cluster boundaries. In general, the SVC algorithm involves three main steps [2]: a) finding the hyper-sphere by solving the Wolfe dual optimization problem, b) identifying the clusters by labeling the data points with cluster labels, and c) searching a satisfactory clustering outcome by tuning kernel parameters. SVC algorithm is to look for the smallest sphere that encloses the images of data points in the feature space. This sphere is then mapped back to the data space, where a number of contours which enclose the data points are formed. These contours are interpreted as cluster boundaries. In general, the SVC algorithm involves three main steps [2]: a) finding the hyper-sphere by solving the Wolfe dual optimization problem, b) identifying the clusters by labeling the data points with cluster labels, and c) searching a satisfactory clustering outcome by tuning kernel parameters. In earlier research work of Wang and Chiang [3], they have developed an effective parameter search algorithm to automatically search suitable parameters for the

SVC algorithm. However, there is a common agreement in SVC research community—solving the optimization problem and labeling the data points with cluster labels are time-consuming in the SVC training procedure. The above limitations make the SVC algorithm inapplicable for large datasets [4]. From literature, we found that many research efforts have been conducted to improve the efficiency of cluster labeling. Because the computation of cluster labeling is considerably expensive, many researchers have engaged in reducing time complexity of this aspect. Yang *et al.* [25] used proximity graphs to model the proximity structure of datasets. Their approach constructed appropriate proximity graphs to model the proximity and adjacency. After the SVC training process, they employed cutoff criteria to estimate the edges of a proximity graph. This method avoids redundant checks in a complete graph, and also avoids the loss of neighborhood information as it can occur when only estimating the adjacencies of support vectors. Lee and Lee [7] created a new cluster labeling method based on some invariant topological properties of a trained kernel radius function. The method they proposed consisted of two phases. The first phase was to decompose a given data set into a small number of disjoint groups where each group was represented by its candidate point and all of its member points belong to the same cluster. The second phase was then to label the candidate points. Nath and Shevade [2] presented a novel approach that increases the efficiency of the SVC scheme.

The geometry presented in the clustering problem was exploited to reduce the training data size. Their experiments showed that the pre-processing procedure drastically decreased the run-time of the cluster algorithm. However, different pre-specified parameters could produce totally different clustering results. Wang and Chiang in 2008, proposed an efficient pre-processing procedure for SVC. This procedure reduces the size of the training dataset. Then SMO algorithm is applied on the reduced training set [4]. HRE [2] and SNN [4] based data pre-processing techniques used for SVC to reduce size of training dataset result in loss of data. The rest of paper is organized as follows. In Section II discuss problem in support vector clustering. The Section III discusses technique of support vector clustering. Followed by a conclusion in Section VI.

II. Problem in Support Vector Clustering

Support Vector Clustering (SVC) algorithm has two main steps: 1) SVC training and 2) Cluster labeling [39]. Solution of optimization problem and labeling the data points with cluster labels is time consuming in SVC algorithm. This limitation of SVC makes them inefficient for large datasets. There are many techniques exist in literature to reduce time complexity of cluster labeling step such as complete graph (CG) strategy [1], modified complete graph (SVG) strategy [9], proximity graph modeling [25], 2-phase cluster labeling strategy [7]. Only little efforts have been made to improve the efficiency of SVC training step. Due to noisy datasets accuracy and efficiency of clustering algorithms get decreases. Some of the data preprocessing procedures exist for SVC are: 1) Heuristics for Redundant-point Elimination (HRE) [2], and 2) Data preprocessing based on Shared Nearest Neighbor Algorithm [4]. Noise reduction and outlier detection based on SNN technique is efficient process, but this SNN based pre-processing procedure generate result on the consideration of loss of data.

III. Support Vector Clustering Technique

In the process of background and survey found that various method and technique used for the improvement of support vector clustering. Some method and technique discuss here. Kernel-based learning algorithms have become increasingly important in pattern recognition and machine learning, particularly in supervised classification and regression analysis with the introduction of support vector machines. Clustering algorithms are capable of finding clusters with different shapes, sizes, densities, and even in the presence of noise and outliers in datasets. Although these algorithms can handle clusters with different shapes, they still cannot produce arbitrary cluster boundaries to adequately capture or represent the characteristics of clusters in the dataset.

Clustering Techniques

Different approaches to clustering data can be described with the help of the hierarchy shown in Figure 1. At the top level, there is a distinction between hierarchical and partitional approaches (hierarchical methods produce a nested series of partitions, while partitional methods produce only one).

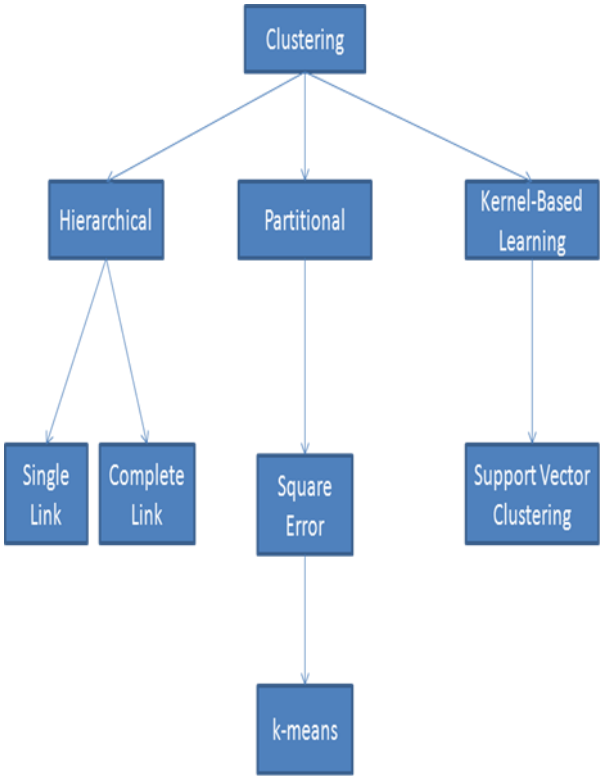

Figure 1: Taxonomy of clustering approaches.

Support Vector Clustering (SVC) involves following steps [2]: It is shown in following figure 2.

1. Data Preprocessing: Eliminates insignificant points and gives reduced training set.
2. Kernel-parameter Tuning: Gives the value of (C, q).
3. Optimization using SMO Algorithm: Solving dual for Lagrange multipliers.
4. Cluster Labeling: Labeling the data points with cluster labels.

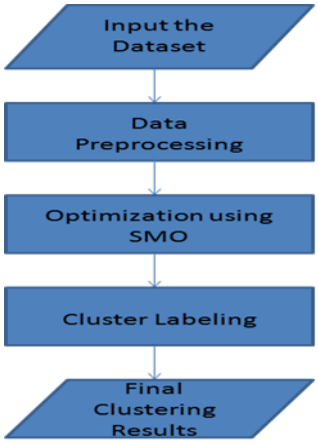

Figure 2: Flowchart of the SVC Procedure
Data Preprocessing Procedures for SVC.

Currently there are two data preprocessing procedures are available in literature for support vector clustering (SVC). These preprocessing techniques remove noise points, outliers and insignificant points which are not important for clustering. They reduce the size of the training dataset. After preprocessing, Sequential Minimal Optimization (SMO) algorithm is applied on the reduced dataset for solving the optimization problem. Next, labeling of each data point with appropriate cluster labels is done using cluster labeling method [2, 4].

HRE-SVC: Pre-Processing using R*-tree

Figure 3 shows clustering with SVC on a synthetic dataset. Note that, NSVs lie inside the cluster boundaries. From the figure it can be seen that the points that are well surrounded by a large number of points from all directions have a high chance of being NSVs. These observations suggest that, we can eliminate points from training data that have following properties [2]:

P1 very large number of neighbors,
P2 neighboring points from all directions.
This elimination process if executed carefully, will not affect the final clustering. Saketha Nath and Shevade's preprocessing method uses these ideas to eliminate redundant data points from the training data. In order to rank points according to P1: for each point, k_1 nearest neighbors are selected and the distance d, between the point and the farthest of the k_1 neighbors is calculated. Greater the value of d, lesser the neighbors it has. In order to rank points according to P2: for each point, the resultant, □, of unit vectors drawn from the point to its k_2 nearest neighbors is calculated. The lesser the value of □, the greater is the possibility that it has neighboring points from all directions. All nearest neighbor queries are processed using R*-tree data structure, queries are preprocessed using R*-tree data structure, which is the most efficient structure known for nearest neighbor queries [2]. For each point in the training set, the values of d and □ are calculated. These values are then normalized, to lie in [0, 1]. The "weight" of each point (quantifies P1 and P2 of a point) is calculated as $\eta d+(1-\eta)$□, where η is a factor in [0, 1]. The points are then listed in decreasing order of weight. Thus, the head of the list will be dominated by points who have very few neighbors or neighbors in few directions, whereas, tail will be dominated by redundant points. Thresholds θ_1, θ_2 ($>\theta_1$) are selected. Points listed above threshold θ_1 are

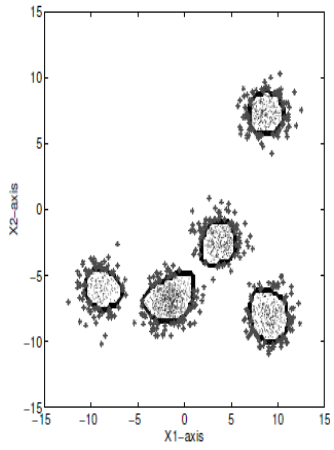

Figure 3: Clustering of a synthetic dataset. Points marked '·' are NSVs and those marked '*' are SVs. Cluster boundaries are the thick contours.

Considered outliers, those between θ_1 and θ_2 are used as training set (non-redundant points) and those below θ_2 are considered as redundant points. A point with P1 and P2 is redundant only in presence of the neighbors. Thus, in the above method some non-redundant points may get eliminated because of the static nature of weight assignment. Ideally, after each point is eliminated from the training set, weight assignment must be redone. However, this dynamic weight assignment method will be inefficient. In view of this, the above method is modified as follows: Weight assignment for all points is done and outliers, reduced training set and redundant points are identified as given above. k_3 nearest neighbors of every non-redundant point are computed. A redundant point is selected from these set of k3 neighbors and sent along with the non-redundant point to the training set (none will be sent if no such point exists). This reduces the possibility of eliminating non-redundant points from the training set. It is easy to see that, the reduced training set size with this method is $2(\theta_2 - \theta_1)$. This pre-processing method will be referred to as "Heuristic for Redundant point Elimination" (HRE). Figure 2.7 shows the result of pre-processing using HRE on a synthetic dataset (m = 500; n = 2; K = 5; θ_1 = 0; θ_2 = 100) [2]. Note that, the HRE works well even when clusters are arbitrary shaped. The pre-processed data is used as the training set in optimization (solved by SMO). The extra effort required for pre-processing using R*-tree is O(m ln m exp(n)). Note that, the heuristic needs to be run only once (as a pre-processing step), and need not be run again during the entire tuning stage or later stages.

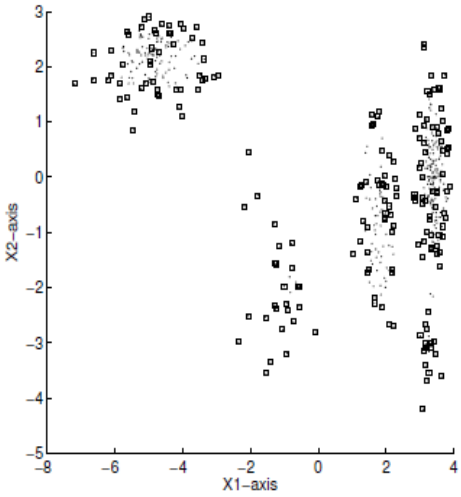

Figure 4: Scatter plot of a synthetic dataset. Pre-processed using HRE. Points marked '·' are redundant points, '▫' are reduced training set points.

Thus, increase in effort due to pre-processing is negligible when compared to decrease in effort at later stages. However, due to the O(m ln m exp(n)) effort in pre-processing, the HRE may not be feasible for very high dimensional datasets [2].

SNN Based Pre-processing Procedure for SVC

Solving the optimization problem and labelling the data points with cluster labels are time-consuming in the SVC training procedure. This makes using the SVC algorithm to process large datasets inefficient [4]. Thus, how to exclude redundant data points from a dataset is an important issue for minimizing the time spent in solving the optimization problem of the SVC algorithm. Researchers challenge in this topic is how to identify insignificant data points so that the removal of these data points does not significantly alter the final cluster configuration. An idea of Wang and Chiang [4] is to eliminate insignificant data points, such as noise and core points, from the training datasets, and use the remaining data points to do the SVC analysis. Due to the size reduction of the training datasets, the computational effort for solving the optimization problem can be greatly decreased. To fulfill the idea, Wang and Chiang first explore the shared nearest neighbour (SNN) algorithm [22, 23] to eliminate noise points. Subsequently, the concept of unit vectors [2] is employed to reduce the core points of clusters and to retain the data points near the cluster boundaries. Based on these two methods, Wang and Chiang developed an efficient data pre-processing procedure for SVC to reduce the size of the training datasets without altering the cluster configuration of the datasets [4].

Cluster Validity Method for SVC

Cluster Validity method for SVC [3] is summarized as follows:

1. Initialize a small value for q and set C= 1.
2. Perform the SVC algorithm to obtain the number of clusters.
3. If the number of clusters < 2, increase the value of q and go to Step 2. Otherwise, go to Step 4.
4. Compute the validity measure index (the ratio V(m)).
5. If the number of clusters ≤ √K, increase the value of q and go to Step 2. Otherwise, if the result of SVC has singleton clusters, decrease the value of C and reset the value of q, and then go to Step 2. If the decrease of C does not change the number of clusters, stop SVC algorithm (C = 1) and go to Step 6. Otherwise, the value of C is identified.
6. Obtain the optimal cluster structure and the final value of q and C.

Figure 2.8 shows the flowchart of the SVC with a cluster validity method.

Cluster Validity Measure with OD for SVC

Several cluster validity indexes have been presented. However, none of them considers the special properties of the SVC algorithm. Many of the validity techniques that compare the inter-cluster versus intra-cluster variability tend to favor configurations with ball-shaped well-separated clusters. Using the existing cluster validity measures for irregularly shaped clusters is problematic because the existing validity measures are not able to measure the distance between two clusters with nonlinearly separable.

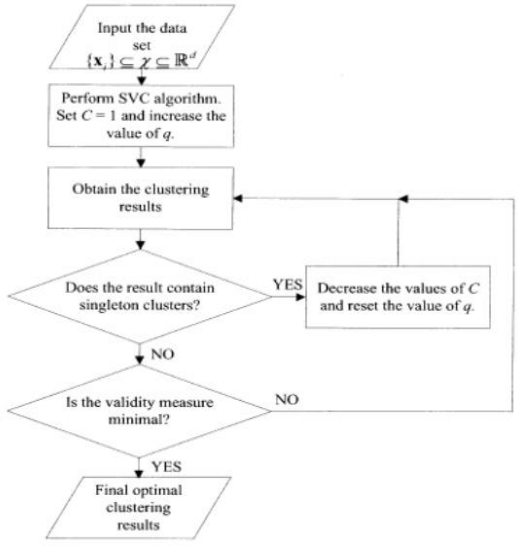

Figure 5: Flowchart of the SVC with a cluster validity method.

Arbitrary shapes. In addition, the performance of these measures usually degrades when the data sets contain noise or outliers, which means that they lack an effective mechanism to deal with noise or outliers. Parameter selection is a critical step for the SVC algorithm. There are two tunable parameters in SVC, namely, q and C. Note that the parameter q is involved only in the Gaussian kernel-based SVC, whereas the parameter C is independent of the chosen kernel. In the supervised support vector learning, it is unknown beforehand which q and C are the best choices for one problem. Consequently, some kind of model selection (or parameter search) must be performed. The goal is to identify suitable q and C so that the cluster configuration can accurately depict the distribution of the data set. To achieve such a goal, cross validations can be engaged to find the best parameters for a given data set. However, it is time-consuming to perform cross validation for large data sets, and what is more, SVC belongs to the category of unsupervised learning, where the class labels are not available in advance. Ben-Hur *et al.* [9] have provided a heuristic rule for choosing these two important parameters. In general, following their rule to make reasonable adjustments for these two parameters may result in desirable clustering outcomes. However, the time-consuming procedure of iterative executions of the SVC algorithm with different parameter selections is necessary for obtaining a desirable outcome. Moreover, varying the value of C to allow for the existences of outliers may increase the chance of preferable contour separations, but it may create subsidiary clusters that hinder the chance for discovering the physical cluster configuration. Hence, the clustering result is sensitive to the value of C, and some trial-and-error efforts are usually inevitable for reaching a desirable outcome when Ben-Hur's heuristic rule is applied. To prevent these two drawbacks while maintaining a minimal number of clusters and assuring smooth cluster boundaries, Wang and Chiang proposed a systematic approach that integrated a new cluster validity measure, an outlier detection method, and a cluster merging mechanism [24].

This algorithm is summarized in the following steps [24]:

1) Initialize a small value of q, and set $C = 1$ and $\gamma = 2$ (or a reasonable value).
2) Perform the SVC algorithm to obtain the number of clusters.
3) If the number of clusters < 2, increase the value of q, and go to Step 2).
4) If the outlier-detection criterion holds, abandon the clustering results, decrease the value of C, fix

the value of *q*, and go to Step 2. Otherwise, go to step 5.
5) If the total number of SVs < 50% of the data points, go to Step 6. Otherwise, abandon the clustering results, decrease the value of *C*, fix the value of *q*, and go to Step 2.
6) Compute the validity measure index (ratio *V* (*m*)).
7) If the number of cluster < $\sqrt{N}$, increase the value of *q*, and go to Step 2. Otherwise, stop the SVC algorithm. The final number of clusters and suitable values of *q* and *C* are identified.
8) Use the cluster-merging mechanism to identify an ideal number of clusters.
The figure shows the cluster validity measure with outlier detection and merging mechanism for SVC.

IV. Conclusion and Future Work

In this paper we review of support vector clustering technique over the improvement of noise and outlier problem faced for data labelling and grouping. Kernel function of support vector cluster is play a important role for data mapping. This paper also describes the procedure for constructing cluster based SVM, i.e. CK-SVM. In this regard we have introduced a cluster based simple and fast training algorithm to solve outliers and computational cost problem. In addition, CK-SVM has provided efficiency for fast classification and continuous outputs via weighted distances for multiclass classification. Outlier detection encompasses aspects of a broad spectrum of techniques. Many techniques employed for detecting outliers are fundamentally identical but with different names chosen by the authors. For example, authors describe their various approaches as outlier detection, novelty detection, anomaly detection, noise detection, deviation detection or exception mining. in future we minimised the outlier and noised in support vector clustering using feature optimisation technique. For the optimisation of feature we used genetic and ant colony algorithm.

References

[1] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, "A Support Vector Clustering Method", In Proc. of Int. Conf. on Pattern Recognition, 2000, pp. 724-727.
[2] J. Saketha Nath, S.K. Shevade, "An Efficient Clustering Scheme Using Support Vector Methods", Pattern Recognition, 2006, 1473-1480.
[3] J. S. Wang, J. C. Chiang, "A Cluster Validity Measure with a Hybrid Parameter Search Method for Support Vector Clustering Algorithm", PatternRecognition, 2008, pp. 506-520.
[4] J. S. Wang, J. C. Chiang, "An Efficient Data Preprocessing Procedure for Support Vector Clustering", Journal of Universal Computer Science, 2009, pp. 705-721.
[5] A. Jain, M. Murty, P. Flynn, "Data Clustering: A Review", ACM Computing Surveys,1999, pp. 264-323.
[6] J. C. Platt, "Fast training of support vector machines using sequential minimum optimization", Advances in Kernel Methods Support Vector Learning, 1998, pp. 185-208.
[7] J. Lee, D. Lee, "An Improved Cluster Labeling Method for Support Vector Clustering", IEEE Trans. Pattern Analysis and Machine Intelligence, 2005, pp. 461-464.
[8] K. Jong, E. Marchiori, and van der Vaart, "Finding Clusters using Support Vector Classifiers", ESANN Proceedings. Europian Symposium on ANN, 2003, pp. 223-228.
[9] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, "Support Vector Clustering", Journal of Machine Learning Research 2, 2001, pp. 125-137.
[10] C. Blake, E. Keogh, C. Merz, "UCI Repository of Machine Learning databases", Department of Information and Computer Sciences, University of California, Irvine, 1998.
[11] Tom Mitchell, "Machine Learning", McGraw Hill, Computer Science Series. 2005, Page no. 2-4, 81-95, 238-245.
[12] Nils J. Nilsson, "Introduction to machine learning", 1997, page no 1-15.
[13] Arun Pujari, "Data Mining Concepts", page no 2-25.
[14] Steve Gunn, "Support Vector Machine for Classification and Regression", Technical Report, Faculty of Engineering, Science and Mathematics, School of Electronics and Computer Science, 10 May 1998, Page No. 2-8, 19-23.
[15] Ethan Alpayadin, "Introduction to machine learning", MIT press Cam-bridge, 2005.
[16] Christopher M. Bishop, "Pattern Recognition and Machine Learning", Springer Publication, Singapore, 2006, Page no. 1-3, 308-320.
[17] Ian H. Witten, Eibe Frank, "Data Mining-Practical Machine Learning Tools and Techniques", Morgan Kaufmann Publishers, Second Edition, 2005, pp. 7-9.
[18] V. Vapnik, "The Nature of Statistical Learning Theory" Springer, N.Y., 1995, ISBN 0-387-94559-8.
[19] J. P. Lewis, Tutorial on SVM, CGIT Lab, USC, 2004.
[20] E. Osuna, R. Freund, and F. Girosi, "Support Vector Machines: Training and Applications", A.I. Memo No. 1602, Artificial Intelligence Laboratory, MIT, 1997.
[21] Jiawei Han and Micheline Kamber, "Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, Second Edition, 2006, pp. 355.

[22] L. Ertoz, M. Steinbach, V. Kumar, "Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data", In Proc. of SIAM Int. Conf. on Data Mining, 2003, pp. 1-12.

[23] R. A. Jarvis, E. A. Patrick, "Clustering Using a Similarity Measure Based on Shared Nearest Neighbours", IEEE Trans. Computers, C-22, 11, 1973, pp. 1025-1034.

[24] J. S. Wang, J. C. Chiang, "A Cluster Validity Measure with Outlier Detection for Support Vector Clustering", IEEE Trans. Systems, Man, and Cybernetics-Part B, 38, 1, 2008, pp. 78-89.

[25] J. Yang, V. E. Castro, S. K. Chalup, "Support Vector Clustering Through Proximity Graph Modeling", In Proc. of 9th Int. Conf. on Neural Information Processing, 2002, pp. 898-903.

[26] D. Tax and R. Duin, "Support vector domain description", Pattern Recognition Letters, vol. 20, 1999, pp. 1191-1199.